Learn to solve problems: a virtual ethnographic case study of learning in a GNU/Linux Users Group

Huysman, M.H.; Lin, Y.

published in
The Electronic Journal for Virtual Organizations and Networks
2006

The Electronic Journal for Virtual Organizations and Networks

Volume 7, December 2005

LEARN TO SOLVE PROBLEMS: A VIRTUAL ETHNOGRAPHIC CASE STUDY OF LEARNING IN A GNU/LINUX USERS GROUP

Marleen Huysman and Yuwei Lin

Vrije Universiteit Amsterdam, Department of Business Administration, Amsterdam, The Netherlands.

ABSTRACT

Linux User Groups (LUGs) epitomise on-line communities where members sharing the same interests learn mutually through engaging in collective practices. LUGs serve as a platform for interested Linux users to solve each other's problems (mainly technical) through interacting with each other on-line with the help of Internet technologies such as mailing lists and Internet Relay Chat (IRC). Although members also meet face-to-face in the physical environment, we have observed that their on-line interactions play a more important role in their mundane problem-solving processes. Thus, we study the digital narratives encoded in the mailing list of York Linux User Group (YLUG) to understand how Linux users communicate with each other virtually to solve their problems. The research is framed under the social worlds theory in order to understand the dynamics and complexities in the problem solving process better. In contextualising and analysing these digital narratives, we find that problem-solving involves continuous negotiations and re/definitions on a locally-found problem. We also find that on-line communities without a strict memberships such as YLUG facilitates active cross-boundary learning and knowledge sharing between interested Linux users coming from different social worlds. This type of cross-boundary learning is quite distinctive compared with traditional learning in a physical environment or even with other on-line communities with strict memberships. While this type of on-line learning communities is shown to be advantageous in some way, it requires further studies in order to understand the disadvantages derived from such loose coordination. As such, we could capitalise on different types of learning for different purposes.

1 INTRODUCTION

GNU/Linux (hereafter 'Linux') is the most well-known open source operating system currently. For the past decades, we have observed an increasing number of Linux users groups (LUGs) established around the world to bring together those who have adopted Linux or are interested in learning about Linux, and to help each other with some aspect of its use or adoption. These on-line communities create a platform for Linux users coming from diverse social worlds to share knowledge and experience on Linux. Apart from serving as a local reference point piling up published reference materials ranging from books and print journals to electronic writings and making them available on their web sites, LUGs usually have mailing lists and Internet Relay Chat (IRC) channels dedicated to the discussion of Linux. Moreover, members of local LUGs often meet up in real life to know each other better, although their on-line activities appear to be more active than their face-to-face communication. Through interaction and negotiation on-line and off-line, LUGs members mutually learn to solve problems that they encounter in the daily computing activities.

However, this problem-solving process is not a straightforward ask-and-answer one-way activity. It involves various heterogeneous and contingent factors concerning cognition, social skills and technical expertise. Unlike traditional learning that 'connotes a one-way process of submitting to external authority', the learning process in LUGs, particularly through the Internet, symbolises 'a mutual process of coordinating perspectives, interpretations, and actions so that [members] realize higher goals' (Wenger, 2003, p. 79). More importantly, information and knowledge is usually shared in an informal environment on the mailing lists or during chat when drinking in a pub. Local knowledge and tacit skills is shown to be as of important in the Linux development and adoption.

To bring the importance of tacit knowledge sharing to the fore, we have conducted an on-line ethnographic study on the interactions of the members of a LUG: the York Linux User Group (YLUG). We analysed knowledge sharing processes related to the interactions of knowledge exchange and the processes of mutual help and mutual learning. By analysing dialogues between Linux users, we demonstrate the values of collective learning (i.e. negotiation) and tacit knowledge in problem–solving process among users and designers as well as amongst amateurs and experts. Although these shared practices take place amongst people from different social worlds, we will argue that as a result of shared tacit knowledge and boundary objects, innovation and objectification of knowledge can occur.

This analysis will help us address mutual learning through cross-boundary collaborations in an on-line community. In investigating how hands–on practices and ongoing negotiation help to construct software problems and solutions, we aim to understand the process of translating local knowledge into formal expertise. While Free/libre open source software (FLOSS) is gaining unprecedented recognition at an accelerating pace, and more institutional resources are made available - noticeably from national agencies and industry - it

is crucial to get a genuine in-depth insight into the FLOSS development, on which this paper will shed some light.

2 DEMOGRAPHY OF THE YORK LINUX USER GROUP (YLUG)

The York Linux User Group (YLUG)[1] is an informal organisation without restriction of membership. It is geographically based around the University of York, but is open to everyone, and the membership includes a good proportion of individuals not affiliated with the University. There are about 80 group members from different sectors (government, industry, academic institutes) living in York and the surrounding areas. Members gather because they share the same interests (mostly technical) in Linux. This group of enthusiasts engages with a wide range of levels of expertise and experience. Though meetings are held once every two weeks during the University's term, the operation of the group depends heavily on the Internet. Hence, there is a strong tendency for the membership of the group to act as a virtual organisation.

The YLUG mailing list works actively and topics vary ranging from hardware information, product reviews, security news, exchanges of tricks or tips and so on. A study of the Finnisch LUGs (FLUGs) (Silvonen, 2002) illustrates four main mainly four types of discussions going on FLUG's mailing list[2]. (Silvonen, 2002) breaks down the activities on the list into the following categories: asking for help (62.4%), business about FLUG (17.6%), Linux–focused (12%), and other issues (8%). While it may be arguable that Silvonen's categories of traffic on the list are over–simplified, his observations prove a kind of commonality amongst LUGs around the world. The YLUG, which is heavily technique–oriented, appears as a virtual organisation for exchanging information and sharing knowledge, where networking at the local level is essential. As Silvonen notes, "The community is the result of continuing joint activity on the Net, of conflict and negotiations by which the identity and borders are created" (Silvonen, 2002).

As virtual organisations, LUGs signify social and organisational innovation as well as technical creativity within virtual environments. The essential dimensions of this organisational form include "(a) networking through the use of ICTs, (b) restructuring into a decentralised network of companies, and (c) building a team culture" while "the boundaries of the organisation and the units that compose them are becoming increasingly permeable" (Dutton, 1999, pp. 474-475). Mainly interacting through ICTs, group members are able to distribute to and link their local activities and information globally, whilst conventional face–to–face communication still plays an important role in the group, building cohesion and trust. As Silvonen remarks, "The Linux community is a complicated network of different communities. Although the Internet is the essential tool for these, they are also anchored in local social activities" (Silvonen, 2002).

[1] http://www.york.lug.org.uk/intro.shtml
[2] With 1,800 subscribers and 241 individual threads consisting of 3,248 topics

3 SOCIAL WORLDS THEORY AND VIRTUAL COMMUNITIES

We would like to emphasise that LUGs, like many other on-line communities, are not homogeneous, spontaneously-emergent, harmonious 'networked community' (Wellman, 1999) upheld by the Internet technologies. A user group would not be formed just because there were Linux users around. It requires much effort, including connecting these people, facilitating their communications, identifying and achieving mutual goals, which centre on solving technical problems in this case. Although the Internet activities have clustered individuals together as virtual communities (Rheingold, 1993), these virtual spaces “differ in their rules about the conduct of speech as well as the speech” (Mason, 1996: 5). Moreover, not only does each virtual community behave differently in their distinctive ways, diversity exists within the community as well. In the case of LUGs, members, though sharing the same interest in Linux and the collective practice in using Linux, give different meanings to Linux and the practice of using it. Consequently, we strengthen the heterogeneity and diversity in the Linux 'social world', where local knowledge and situated knowledge derive from identities and commitments largely developed through prolonged interaction toward shared, yet continually emergent, goals (Lin, 2004).

We have grounded our analysis under the social worlds theory (SWT) (Bowker & Star, 1999; Clarke, 1991; Fujimura, 1992; S. L. Star & Griesemer, 1989). Social worlds can be seen as – often overlapping - reference groups, such as an occupation that generate shared perspectives, which form the basis for collective action (Shibutani, 1955). The notion of social worlds has been available in the sociological literature for many years (e.g. Park, 1952; Shibutani, 1955; Strauss, 1978). Its roots lie in the concern of the symbolic interactionism with the making of communities and organizations. In this tradition, social groups of various kinds structurally situate individuals in society (Clarke, 1991). One can think of social worlds as for example the world of the deaf, the advertisement world, the world of motorcyclists, the guy world, etc. An important feature of social world is that they are not bounded by geography or formal memberhsip but “the limits of effective communication” (Shibutani, 1955). A social world is an interactive unit, a “universe of regularized mutual response, communication or discourse (ibid.). As a result, they influence the meaning that people impute on events. Social worlds consciously or unconsciously inform people what knowledge is important and what knowledge is not (Mead, 1935).

The concept of social worlds is similar to that of a community of practice (Lave & Wenger, 1991). Both are highly fluid and emergent structures consisting of individuals with a shared collective interests. However, SWT stemming from the tradition of science and technology studies (STS) looks more at the interplay between artifacts and actors (individuals or organisations). It also stresses the importance of boundary objects, negotiation and the fluidity/mobility of actors. SWT allows researchers to study the network formed around a boundary object attracting diverse actors to negotiate with it. In other words, SWT is useful in analysing a complex and dynamic arena where many cross-boundary activities exist.

In the case of YLUG, since most of the on-line conversations are about problem-solving, these problems, mostly technical-oriented, serve as boundary objects to engage diverse actors on cross-boundary collaboration. In this paper, we explore how people negotiate to solve the boundary problem from conceiving a problem to providing a solution to it. Through looking at the problem-solving process, we try to highlight multiple visions and means of achieving problem-solving by attempting empirically to view the world in the actors' own terms. This perspective based on SWT, would help us understand how diverse actors communicate over the always emergent, negotiated and situated concepts through engaging on the collective practices. With such a view of 'social worlds', a community, in our case the YLUG, is not pre-given and presented as 'a matter of fact'; a community is an ecological entity inhabiting diverse actors interacting with each other.

4 VIRTUAL ETHNOGRAPHY

As Silvonen (2002) points out, activities of LUGs heavily dependent on Internet technologies within virtual spaces. As such, not only can users talk to each other, but they can also build their own virtual spaces. Therefore, virtual ethnography would be the most useful research approach to understand the members' distinctive interactions that involve on-line mutual learning and knowledge-sharing. To approach our fieldwork, the first author of this paper has joined the mailing list of YLUG and followed the discussions on-line continuously for 3 years. In line with qualitative methodology, the virtual ethnography helps understand 'how' mutual learning and knowledge sharing take place in YLUG.

The researcher was aware that a lot of the members of the group were also communicating via personal e-mail. This brings up the problem that by just observing the mailing list, we might miss out the contingency emerging from these private contacts. However, the researcher did not just read the mailing list archives. She followed the virtual conversation very closely and therefore did not distance herself too much from the development of the community. That said, even in virtual ethnography, there is no substitute for 'being there', even if 'there' simply means sitting at your computer looking at the web site on a regular basis (Preece & Maloney-Krichmar, 2003; Ruhleder, 2000).

Below, we present two examples of mutual learning amongst the YLUG members to illustrate how actors from different social worlds solve problems through negotiation and boundary crossing.

5 MUTUAL LEARNING

Although the standard model of organisational learning perceives the process as autonomous and one-directed, a single learner learns from an exogenous environment, some maintain that organisational learning often involves mutual learning (Huysman, 1999). Learning in one part of the organisation interacts with the learning in other parts; learning in one organisation interacts with learning in other organisations (e.g. Levinthal & March, 1993; Lounamaa &

March, 1987). In this research, we analyse how the learning within YLUG occurs among members of different social worlds (e.g. academics, business, government, experts, amateurs, etc.). As an extension of their mutual learning, their cross-boundary learning is embedded on their on-line activities on the mailing list. The YLUG mailing list serves helps the members discuss a wide range of disciplines in an open learning environment across boundaries. A cross boundary learning community is thus created through the members' on-line interactions.

Most members in the YLUG are experienced computer experts and they all know this. Face-to-face meetings also deepen their understanding of each other's biographical background. The reciprocation and expectation between them facilitates trust and cooperation and vice versa. With a shared interest in the Linux operating system, and a common believe in 'free' sharing and contributing, people from different sectors and areas are brought together in computing practices as computing learners, businessmen, engineers and scientists - 'sometimes as insiders, sometimes as outsiders" (S. L. Star, 1995). As the message shows below, a skilful user would act like an outsider when coming across with a certain type of question:

> *I have a large pile of (scanned) photos from a holiday. A lot of these are panoramas which I want to join together to make one big picture. Anyone recommend any software for doing this easily? If your answer is the GIMP then please point me at some instructions since I had a f[r]ustrating time with it!(YLUG032503)*

The author of this message was a systems security advisor at the Computing Service department. The community regards him as very knowledgeable in computing. Nevertheless, his message presented him as an 'outsider' of processing graphical data. Two messages follow up.

> (1)
>
> *One way to do it in the gimp:*
>
> *Open one of your pictures, note the size of the image.*
>
> *Open a new blank image and create it of size equal to 3x (or however many photos you want to join) the width. Allow 20% or so extra on the h[ei]ght as well.*
>
> *Then open the rest of your original images. Copy them and paste them into the new image. Make sure you've got the ones you paste first lined up before you add others.*
>
> *When you're either save, or adjust canvus size, or just drag a box around them and copy. Then create new and it'll automatically be the right size for you to paste your copied image into.*
>
> *Save however you like.*
>
> *(YLUG042503)*

The author of this message was a student at the chemistry department who presumably possesses less knowledge than a staff member in computer service. He acted like an insider here because he was skilful at this specific software application. Another message follows:

> *Sounds mostly right to me. You may find lining things up easier if you load each image into a different layer, then you can set each layer to be pa[r]tial[l]y transparent. I hope you've got a fast machine.*
>
> *(YLUG052503)*

This author was an employee in a tele-computation company. From his message, he acted as a more experienced insider than the student: “Sounds mostly right to me.” And he added some small points to contribute his knowledge.

This thread is chosen for our analysis because it shows how people from different social worlds can contribute to each other's learning. It also shows that the knowledge hierarchy in computing has been broken down in this knowledge-sharing and mutual-learning activity: a computer expert learned from an amateur student. Local participants, even without meeting with each other in real world, bring their knowledge and experience to bear on local problems and issues in learning across different boundaries. For instance, from this thread, we see how non-canonical practices based on personal experiences have been objectified and distributed as canonical practice and a source of innovation (Brown & Duguid, 1991) that can be used by an experienced staff for his hobby— compiling family holiday photos.

6 SOLVING PROBLEMS THROUGH NEGOTIATIONS

As mentioned earlier, the most common topic on the mailing lists of LUGs is asking for help. Although the transparent feature of FLOSS helps open the black box of the software technologies, not everybody could understand what is inside this black box. Getting onto www.kernel.org and downloading the source code of the Linux kernel programme hardly makes any sense for someone with little knowledge of the basic programming language. This is not only because one lacks competence to read code, but also that one may not have enough experience (tacit knowledge) to find out the wanted information from thousands of lines of code. Therefore, there is an argument that FLOSS is too expert-centred to be user-friendly, and only the software competent can use them. Moreover, there are not as many manuals available as those for Microsoft Windows and its peripheral systems. Documentation of Linux, mostly based on contributions from users in the community, is not complete either. At this point, LUGs serve as a channel for users to help each other. Instead of mainstream problem-solving techniques widely used in industries, the practices of LUGs often illustrate ancient trial-and-error innovative methods. These shared problem solving practices enhances innovation and identity formation of the members (i.e. being the member of YLUG and a Linux user) and the boundary of the community (i.e. belonging to YLUG).

The following piece of thread deals with the common situation of incompatible hardware. Hardware problems are very common in mundane computer activities. To solve hardware problems, it requires not only technical expertise but also tacit-knowledge because the problem is so local (depending e.g. which machine one uses) that one cannot deal with it unless one has heard of it or come across to it before. To share the tacit knowledge to solve hardware problems, the discussion usually involves a process of negotiation on re/definition of

a local problem. The tread below illustrates this negotiation process and how a problem is recognised, re/defined, and solved. Although this first message of the thread is rather long, it shows how a question on the mailing list is presented and how a problem is conceived locally.

> *Trying to install mplayer and finding my way through a maze of dependencies which comes as no surprise to me. Have got so far but have now run into the following problem.*
>
> *When I try to install the rpm I get:*
>
> *[root@craig mplayer]# rpm -ivh --nodeps mplayer-0.90-0.rc4.4mdk.i586.rpm*
> *Preparing... ### [100%]*
> *Segmentation fault*
>
> *No matter what I rpm I try to install it fails. Even with the --force option it fails.*
>
> *Trying the above but with two v's (lots ugly debugging information) gives:*
>
> *[root@craig mplayer]# rpm -ivvvh --nodeps mplayer-0.90-0.rc4.4mdk.i586.rpm*
> *D: ============== mplayer-0.90-0.rc4.4mdk.i586.rpm*
> *D: Expected size: 2437862 = lead(96)+sigs(241)+pad(7)+data(2437518)*
> *D: Actual size: 2437862*
> *D: opening db environment /var/lib/rpm/Packages joinenv*
> *D: opening db index /var/lib/rpm/Packages create mode=0x42*
> *D: locked db index /var/lib/rpm/Packages*
> *D: added binary package [0]*
> *D: found 0 source and 1 binary packages*
> *D: ========== recording tsort relations*
> *D: ========== tsorting packages (order, #predecessors, #succesors, tree, depth)*
> *D: 0 0 1 0 0 mplayer-0.90-0.rc4.4mdk*
> *D: installing binary packages*
> *D: opening db index /var/lib/rpm/Name create mode=0x42*
> *Preparing... D: opening db index /var/lib/rpm/Basenames create mode=0x42*
> *### [100%]*
> *D: Expected size: 2437862 = lead(96)+sigs(241)+pad(7)+data(2437518)*
> *D: Actual size: 2437862*
> *D: install: mplayer-0.90-0.rc4.4mdk has 154 files, test = 0*
> *Segmentation fault*
>
> *Any ideas?*
>
> *(YLUG022003)*

No one on the list really knows what's going on there unless one has encountered this problem before. So the following message tried to 'guess' the cause of the problem:

> *Wild random guessing time... What cpu is in that machine?*
>
> *(YLUG122003)*

A member tries to explain the fault by giving reasonable explanation rather than randomly guessing:

> *Outside the obvious "download a new version of RPM and put it in by hand", segfaults are oftentimes caused by buggered memory...*

(YLUG222003)

My first suggestion would be "sod the RPMs and install mplayer from source" - failing that, have you tried running memtest?

(YLUG322003)

The author of the original message tries all methods and responds:

[my cup is] Athlon XP 2100+ [the] memory does serve.

(YLUG422003)

It doesn't matter what rpm I try and install, I seem to the get the same result. So that rules out redownloading it.

(YLUG522003)

Installing from source in't a problem, it's just why its suddenly happeningthat all.

Never heard of memtest but I'm going to try and find it now and see what happens when I run it.

(YLUG622003)

Apparently he is confused about the situation and still desperate for other methods. Here is something new for him: 'memtest'. Just when he was about to try the new method 'memtest', another suggestion came in:

> Never heard of memtest but I'm going to try and find it now and see what

> happens when I run it.

http://www.memtest86.com/

Leave it running for as long as you can, preferably overnight at the veryleast - if you've got dodgy RAM then it'll pick it up.

(YLUG722003)

One more contribution:

> > [root@craig mplayer]# rpm -ivh --nodeps mplayer-0.90-0.rc4.4mdk.i586.rpm
> > Preparing... ##
> > [100%]
> > Segmentation fault
> > <snip>
> > ---
*You rpm db is ****ed. Try*
 # rpm --rebuilddb

first. If that doesn't work, have a look under /var/lib/rpm
for any tmp files that are hanging aruond. Also remove /var/tmp/rpm.*

Sometimes old lock files get left and this causes this behaviour.
Unfortunatley I can't remember where the files get left or what they
are called but that's what caused this for me last time.

If you do this, then do a rebuild again afterwards.

(YLUG822003)

Unfortunately, the last method didn't work either. And the original author decides to try 'memtest', something new for him.

> > Never heard of memtest but I'm going to try and find it now and see what
> > happens when I run it.
>
> http://www.memtest86.com/
> Leave it running for as long as you can, preferably overnight at the very
> least - if you've got dodgy RAM then it'll pick it up.

I'll try this but it may be a while before the results get back for obvious reasons.

(YLUG1022003)

And the result appears to be unpleasant.

Ok the results are in and they are not good :-(

It appears that one of the sticks of RAM is broken at 27.5MB and 148MB.

Thanks for that suggestion. What I need to test now is if it will install with just the good stick or RAM in or is there going to be more problems. Some thing I will have to test at the weekend when I have more time.

What I don't understand is how it managed to install about half of the rpm's then suddenly start having this 'Segmentation fault' error. Correct me if I'm wrong but RAM doesn't just stop working like that.

(YLUG1122003)

The last message of this topic tries to make sense of the whole thing.

> Thanks for that suggestion. What I need to test now is if it will install
> with just the good stick or RAM in or is there going to be more problems.
> Some thing I will have to test at the weekend when I have more time.

If you're feeling really cheap, look up online for the Badram kernel patch. http://rick.vanrein.org/linux/badram/

IIRC, it's kind of a sister project to memtest86...

> What I don't understand is how it managed to install about half of the rpm's
> then suddenly start having this 'Segmentation fault' error. Correct me if
> I'm wrong but RAM doesn't just stop working like that.

OK. Well;
1) RAM /does/ just stop working like that
2) Because linux is pretty clever, which ram you're using changes from moment to moment; segfaults with bad ram are always completely random and annoying.

(YLUG1222003)

As discussed earlier, the thread is chosen because it shows how a solution to a problem is negotiated through diverse people, with different competences/expertises and experiences, guessing what is the cause of the perceived problem. It may require further clarification from the original author of the first message. It also may require much trial-and-error. This thread shows such a distinctive way of interacting with each other in a problem-solving process.

As shown in this thread, after exchanging many messages on the YLUG mailing list, the problem turns out to be a hardware problem, one of the main challenges that software engineering faces. As written in one software engineering textbook, software engineering faces the heterogeneity challenge to operate as distributed systems across networks that include different types of computers and with different kinds of support systems (Sommerville, 2001, p. 13). Since software is a set of instructions detailing the operations to be performed by the computer, inevitably, the design of the hardware shapes the development of the software (Peléz, 1988, p. 2). The heterogeneity challenge, in other words, is the challenge of developing techniques to build dependable software, which is flexible enough to cope with this heterogeneity. As the last message shows, assembling hardware and software together into a single machine is very unpredictable, at least at the current stage. In order to understand this locally defined problem, the members had to exchange several messages to 'guess' what was the problem and try out different techniques. The solution found in the end is a product of a negotiated order (Strauss, 1978) between people coming from different social worlds.

This suggests that problem-solving and mutual learning is an evolving process of negotiation. Solutions to problems are the boundary object(s) that bring diverse actors together in cooperative pursuits. But this problem-solving process, from a sociological point of view, is not that straight forward; instead, it involves many negotiations, re/definitions on meanings of problems. So whether the members could come to an agreement on identifying/defining a boundary object and whether the communication process goes well would be the key to the solution. Additionally, in the negotiated problem-solving process, actors 'co-construct knowledge and skill, drawing on the social and material resources available to them' (Bannon, 1998, p. 49). The relationships between the members also change subtly through re–formulating concepts of different problems posted to the list. Their practices, both individual and collective, define, mark and identify their status and identity.

The activities of the YLUG correspond with the social learning practices characteristic to communities of practice. Within communities of practice, work, learning and innovation emerge mutually and at the same time (Brown & Duguid, 1991). Learning and action are "situated, and (..) work is accomplished via artefacts, in conjunction with others (L. Suchman, 2000, 2001) . The actors who are more likely to provide valid information normally have a superior competency in speaking fluent computing languages and communicating with their colleagues, be they local or virtual. Additionally, innovation is embedded in their everyday, hands–on computing practices.

7 CONCLUSION

We have presented how mutual-learning take place in YLUG, an on-line LUG, through negotiating problems and solutions based on the members' collective practices found in communities where actors interplay. In this problem-solving process, the members come from different social worlds to perform cross-boundary and mutually. The YLUG is a social network that epitomises a community–based innovation system. Innovation within YLUG is developed through collective learning, and everyday practices and tacit skills are shared, learned and evolve into more canonical practices. Within such a community of practice, learning and action are situated. Instead of positioning designers and users in two different knowledge frames, this informal organisational learning brings the members into a common platform where they encounter and interact directly. This platform, mainly in a digital form, serves to build up a community where experts and amateurs share the same interests and act collectively. Engaging in such a community of practices becomes a unique cultural experience for both experts and amateurs. Through mutual learning, the peripheral participation in the FLOSS community makes the software innovation process transparent and permits diverse innovators to engage in it. Creativity and culture thus emerge in this community.

This research contributes to present research on the FLOSS development in that it departs from the more traditional viewpoints on the phenomenon. Usually open source software communities are studied in order to understand what motivates people to contribute freely, or to understand the standardisation issues. These studies typically make use of surveys and interviews and as a result ignore to look into the daily practices that contribute to the software development. The virtual ethnography that is used as a research methodology for this study enables researchers to open this black box of the FLOSS development. It has shown that software development takes place by means of problem solving between actors from different social worlds. Coming from different backgrounds, different experiences of these diverse actors do not limit their mutual learning processes when they meet mainly on-line.

The downside of ethnographic research is that reporting about the studies can be a lengthy endeavour. We have therefore decided to present only two examples of mutual learning. We do hope however that these two illustrations give an idea of how actors jointly learn from each other. Future ethnographic research on the communication between professionals in virtual organisations other than FLOSS will show whether these mutual learning processes, as well as the importance of boundary objects in order for actors from different social worlds to join forces and help each other, are of a more general nature.

REFERENCES

Bannon, L. J. (1998). Computer supported collaborative working: Challenging perspectives on work and technology. In R. Galliers & W. Baets (Eds.), *Information technology and organizational transformation: Innovation for the 21st century organization*. Chichester: Wiley.

Bowker, G. C., & Star, S. L. (1999). *Sorting things out: Classification and its consequences*. Cambridge, MA: The MIT Press.

Brown, J. S., & Duguid, P. (1991). Organizational learning and communities of practice: Towards a unified view of working, learning and innovation. *Organization Science, 2*(1), 40-57.

Clarke, A. E. (1991). Social worlds/arenas theory as organizational theory. In D. Maines (Ed.), *Social organization and social processes: Essays in honour of andelm I. Strauss* (pp. 119-158). New York: Aldine Gruyter.

Dutton, W. H. (1999). The virtual organisation: Tele–access in business and industry. In G. DeSanctis & J. Fulk (Eds.), *Shaping organisation form: Communication, connection and community*. London: Sage.

Fujimura, J. H. (1992). Crafting science: Standardized packages, boundary objects, and translation. In A. Pickering (Ed.), *Science as practice and culture*. Chicago: The university of Chicago Press.

Huysman, M. H. (1999). Balancing biases, a critical review of the literature on organizational learning. In M. Easterby-Smith, J. Burgoyne & L. Araujo (Eds.), *Learning around organizations: Developments of theory and practice*: Sage Publications.

Lave, J., & Wenger, E. (1991). *Situated learning. Legitimate peripheral participation*. Cambridge: University of Cambridge Press.

Levinthal, D. A., & March, J. G. (1993). The myopia of learning. *Strategic Management Journal, 14*, 95-112.

Lin, Y.-W. (2004).*). Hacking practices and software development: A social worlds analysis of ict innovation and the role of open source software.* Unpublished Doctoral Dissertation, University of York.

Lounamaa, P. H., & March, J. G. (1987). Adaptive coordination of a learning team. *Management Science, 33*(1), 107-123.

Mason, B. L. (1996). Moving toward virtual ethnography. *American Folklore Society News, 25*(2), 4-6.

Mead, G. H. (1935). *Mind, self, and society*. Chicago: The University of Chicago Press.

Park, R. E. (1952). *Human communities*. Clenco, IL: Free Press.

Peléz, E. (1988). *What shapes software development?* (Edinburgh PICT Working paper, number 10.). Edinburgh: Research Centre for Social Sciences, University of Edinburgh.

Preece, J., & Maloney-Krichmar, D. (2003). Online communities. In J. Jacko & A. Sears (Eds.), *Handbook of human-computer interaction* (pp. 596-620). Mahwah NJ: Lawrence Erlbaum Associates Inc. Publishers.

Rheingold, H. (1993). *Virtual communities*. Reading, MA: Addison Wesley.

Ruhleder, K. (2000). The virtual ethnographer: Fieldwork in distributed electronic environments. *Field Methods, 12*(1), 3-17.

Shibutani, T. (1955). Reference groups as perspectives. *American Journal of Sociology, 60*, 562-569.

Silvonen, J. (2002). Linux user groups and the 'linux community', *Second Oekonux Conference, Technische Universität Berlin, (1–3 November)*.

Sommerville, I. (2001). *Software engineering* (6th ed.). Essex: Pearson.

Star, S. L. (1995). Introduction. In S. L. Star (Ed.), *The cultures of computing.* London: Blackwell Publishers.

Star, S. L., & Griesemer, J. R. (1989). Institutional ecology, 'translations,' and boundary objects: Amateurs and professionals in berkeley's museum of vertebrate zoology, 1907 - 1939. *Social Studies of Science, 19*, 387-420.

Strauss, A. L. (1978). Social worlds perspective. In N. Denzin (Ed.), *Studies in symbolic interaction* (Vol. 1, pp. 119-128). Greenwich: CT JAI Pr.

Suchman, L. (2000). Located accountabilities in technology production (in progress). from http://www.comp.lancs.ac.uk/sociology/soc039ls.html

Suchman, L. (2001). *Human/machine reconsidered (a work-in-progress paper under development as the introduction to a 2nd, revised edition of plans and situated actions: The problem of human-machine communication)*: Cambridge University Press.

Wenger, E. (2003). Communities of practice and social learning systems. In D. Nicolini, S. Gherardi & D. Yanow (Eds.), *Knowing in organizations: A practice-based approach*. London: M. E. Sharpe, Inc.